Contemporary Mathematics
Volume 157, 1994

A class of domain decomposition preconditioners for massively parallel computers

PATRICK CIARLET JR. AND GÉRARD MEURANT

ABSTRACT. In this paper we develop and experiment with a domain decomposition preconditioner for the conjugate gradient method to solve linear systems arising from the discretization of elliptic partial differential equations. The method uses a two colors ordering of the subdomains and a coarse grid solve with a seven point scheme like matrix. Numerical experiments show that the number of iterations can be made constant when the mesh size is refined and therefore the method seems well suited for massively parallel architectures with a few hundreds of processors.

1. Introduction

The purpose of this work is to develop and implement iterative methods well-suited for massively parallel architectures in order to solve elliptic problems such as

$$-\text{div}\,(\lambda\,\nabla u) = f \text{ in } \Omega, \quad \text{with } \lambda(x,y) = \begin{pmatrix} a(x,y) & 0 \\ 0 & b(x,y) \end{pmatrix}$$

$$u = 0 \text{ on } \partial\Omega.$$

Here $\Omega =]0,1[\times]0,1[$ and a, b and f are given functions, a and b being non-negative over the domain. We approximate these problems by using a P_1 finite element method with right triangles, leading to a five point centered scheme. By this, we are able to handle problems with discontinuous coefficients without any difficulty, as long as the jumps occur along the sides of the triangles. This gives us a linear system

$$Ax = b.$$

1991 *Mathematics Subject Classification.* Primary 65F10, 65F30.
The final detailed version of this paper will be submitted for publication elsewhere

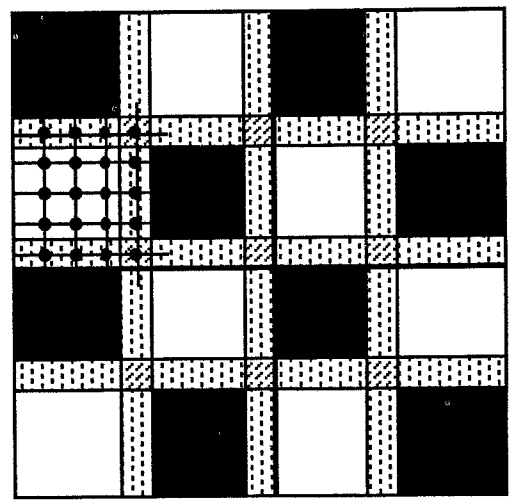

FIGURE 1 A DECOMPOSITION OF THE DOMAIN

We propose to solve this symmetric positive definite linear system by using the Conjugate Gradient method together with a preconditioner designed for efficient use on massively parallel computers with say, more than a hundred processors and a distributed memory. The challenge we are faced with is to have a trade-off between a good convergence rate for the Preconditioned Conjugate Gradient method and a good Mflops rate on the given computers. On the kind of architectures we are interested in, this translates into having a fast code on each processor and minimizing the communications.

2.The Domain Decomposition preconditioners

The construction of the preconditioners is based on partitioning the domain as indicated on Figure 1. This type of partitioning was introduced by Proskurowski and al in [1], [2] and [3]. We consider four types of unknowns:

(1) the nodes in the white boxes (W),
(2) the nodes in the black boxes (B),
(3) the nodes in the rectangular boxes (interfaces) called separators (S),
(4) the nodes in the small striped boxes called crosspoints (C).

Classically we rewrite A as a 4 by 4 block matrix, the indices corresponding to the types previously defined. Thus, we have:

$$A = \begin{pmatrix} A_{WW} & 0 & A_{WS} & 0 \\ 0 & A_{BB} & A_{BS} & 0 \\ A_{WS}^T & A_{BS}^T & A_{SS} & A_{SC} \\ 0 & 0 & A_{SC}^T & A_{CC} \end{pmatrix}.$$

The preconditioner M is defined as

$$M = L\, D^{-1}\, L^T,$$

where D is block diagonal and has the same block diagonal as L and, with the same numbering as before,

$$L = \begin{pmatrix} M_{WW} & & & \\ 0 & M_{BB} & & \\ A_{WS}^T & A_{BS}^T & M_{SS} & \\ 0 & 0 & A_{SC}^T & M_{CC} \end{pmatrix}.$$

At each step of the PCG method, we have to solve a problem with matrix M. Now, the solution of the forward step

$$\begin{pmatrix} M_{WW} & & & \\ 0 & M_{BB} & & \\ A_{WS}^T & A_{BS}^T & M_{SS} & \\ 0 & 0 & A_{SC}^T & M_{CC} \end{pmatrix} \begin{pmatrix} y_W \\ y_B \\ y_S \\ y_C \end{pmatrix} = \begin{pmatrix} c_W \\ c_B \\ c_S \\ c_C \end{pmatrix}$$

is obtained by

1. Parallel solves on the white and the black boxes $M_{ii} y_i = c_i$ $(i = W, B)$.
2. A solve on the interface $M_{SS}\, y_S = c_S - A_{WS}^T\, y_W - A_{BS}^T\, y_B$.
3. A solve on the cross points $M_{CC} y_C = c_C - A_{SC}^T y_S$

Therefore, the total amount of work involved to solve a problem with matrix M is: two parallel solves on the boxes (matrices M_{ii}, $i = W, S$), two solves on the interface (matrix M_{SS}) and one solve on the cross points (matrix M_{CC}). There are a lot of choices for matrices M_{ii}. In a first step, we will consider and implement the following choices:

$M_{ii} = A_{ii}$, $i = W, S$, that is, a direct solver is used for the subproblems.

$M_{SS}^1 = 2\{A_{SS}^{(2)} - A_{BS}^T\, M_{BB}^{-1} A_{BS}\}$, or $M_{SS}^2 = B_{SS} - A_{BS}^T\, M_{BB}^{-1} A_{BS}$.

Here, $A_{SS}^{(2)}$ is the part of the assembly matrix computed from the black boxes, and B_{SS} is equal to A_{SS} except on the diagonal where

$$(B_{SS})_{i,i} = (A_{SS})_{i,i} - |(A_{WS})_{j_i,i}|,\ j_i \text{ such that } (A_{WS})_{j_i,i} \neq 0$$

This corresponds to a "Neumann" boundary condition on the black boxes. In both cases, the coupling with the white nodes is neglected. It is easy to see that if the unknowns in the black boxes are numbered in a "natural" way, then M_{SS} is a block diagonal matrix (each block corresponding to the nodes around a black box) and that the elements of these blocks can be computed cheaply. Indeed, the matrix M_{SS} corresponds to the elimination of the interior nodes of the black boxes. This gives a "clique" that connects all the nodes on the boundary of each box. Thus, the diagonal blocks of M_{SS} are dense matrices.

Finally, we choose $M_{CC} = A_{CC} - A_{SC}^T\, M_{SS}^{-1}\, A_{SC}$. This matrix is sparse and it is also easily seen that it has the structure given by a nine point stencil, two coefficients of which are zero.

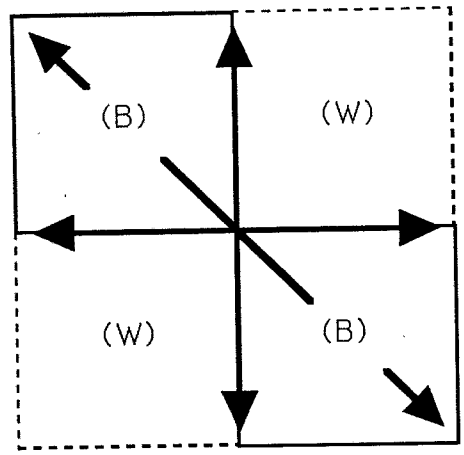

FIGURE 2 NINE-POINT SCHEME WITH TWO NULL COEFFICIENTS

This particular point requires a few more explanations. M_{SS} is block diagonal, each block corresponding to a clique. A cross point is related to exactly two cliques by A_{SC} and its transpose. Thus the term $-A_{SC}^T\ M_{SS}^{-1}\ A_{SC}$ links the cross points as shown in Figure 2. In this paper, we will solve this "coarse grid" system with a Diagonally Preconditioned Conjugate Gradient (DPCG) solver. Further improvements of the method are to use the present algorithm recursively in a multilevel way or to define an approximation of the inverse of the matrix M_{CC} for the cross points. This last method seems better suited for massively parallel architectures.

3. A few numerical examples

We will show some numerical experiments with the implementation of the proposed algorithm on a Sequent S80 parallel computer, with twenty processors, which is a shared memory machine. Here, we will look only at the numerical behaviour of the algorithm and not to its parallel implementation which will be the subject of a forthcoming paper.

The set of different problems is

Problem #1

$a = 1$ in Ω

$b = 1$ in Ω

Problem #2

$$a = \begin{cases} 10 & \text{if } \frac{1}{4} \le x \le \frac{3}{4} \\ 1 & \text{elsewhere} \end{cases}$$

$b = 1$ in Ω

Problem #3

$$a = b = \begin{cases} 10^3 & \text{if } \frac{1}{4} \le x, y \le \frac{3}{4} \\ 1 & \text{elsewhere} \end{cases}$$

We immediately see from the definition of the preconditioner that all the black and white boxes can be solved independently if one box is given to a processor. Therefore, we will study the influence of both the mesh size and the number of boxes for a given mesh size on the rate of convergence. Let h denote the mesh size, n the number of unknowns in each direction ($h = \frac{1}{n+1}$) and n_0^2 the number of boxes. The total number of unknowns will either be 3969, 16129 or 65025. The

corresponding total number of boxes will be 64, 256 or 1024. This translates into the maximum number of processors that can be used in parallel. The stopping criterion is reached when the norm of the residual has been reduced by a factor of 10^{-6} for the PCG method. Tables 1 and 2 givesthe number of iterations for the two choices, M^1_{SS} and M^2_{SS}.

Table 1

Number of iterations for M^1_{SS}

Problem	Pb #1	Pb #2	Pb #3
$n = 63,\ n_0 = 8$	19	26	–
$n = 127,\ n_0 = 16$	20	29	–
$n = 255,\ n_0 = 32$	20	30	–

Table 2

Number of iterations for M^2_{SS}

Problem	Pb #1	Pb #2	Pb #3
$n = 63,\ n_0 = 8$	16	22	17
$n = 127,\ n_0 = 16$	17	24	18
$n = 255,\ n_0 = 32$	17	25	19

The – sign indicates that the PCG method does not converge for Problem #3. Indeed, the preconditioner M^1_{SS} is not a positive definite matrix in this case. In the remaining cases, the number of iterations is always small. There is a further evidence of the effectiveness of the method if one looks at the condition numbers, which remain constant for a given problem and a given preconditioner for all values of (n, n_0), as long as the ratio n/n_0 is constant.

Table 3

Condition numbers

Problem	Pb #1	Pb #2	Pb #3
$\kappa(\{M^1_{SS}\}^{-1}A)$	8	18	–
$\kappa(\{M^2_{SS}\}^{-1}A)$	5	11	8

Both the number of iterations and the condition numbers compare favorably for M^2_{SS}, although exactly the same amount of work is done at each iteration of the PCG method.

Now, we focuse our interest on the DPCG solver for the problems with matrices $M^2_{CC} = A_{CC} - A^T_{SC}\ \{M^2_{SS}\}^{-1}\ A_{SC}$. As the preconditioner is the diagonal of the matrix, this solver is well-suited for an implementation on a parallel architecture. The drawback is that the diagonal is a poor preconditioner from the point of view of condition number, as shown on Table 4.

Table 4

Average number of DPCG iterations for the choice M_{SS}^2

Problem	Pb #1	Pb #2	Pb #3
$n_0 = 8$	16	26	20
$n_0 = 16$	32	49	38
$n_0 = 32$	63	97	74

Interestingly, the average number of iterations (because a certain number of M_{CC} problems are solved) does not depend on the mesh size. It depends only on the number of boxes, or, and that is the same, on $H = \frac{1}{n_0}$, the mesh size for the coarse mesh. Unfortunately, the behaviour of the average numbers of iterations implies that the condition number of the DPCG solver grows like $1/H^2$.

Finally, we note that M_{SS}^2 has many "small" entries. So we try to discard in M_{SS}^2 some of the connections: we choose to neglect some of the fill-ins by value (except on the diagonal). We consider here Problem #1. The number of iterations as a function of the percentage of dropped entries is given in Table 5.

Table 5

Number of iterations for M_{SS}^2

% dropped	0%	50%	75%	100%
$n = 127$, $n_0 = 8$	20	20	21	93
$n = 127$, $n_0 = 16$	17	26	33	117

The worsening is more dramatic when the number of boxes is greater. In some cases, we shall have to use more points or to define other strategies like the one developped by Smith in [Smit91].

4. Conclusion

In this paper, we have presented a preconditioner for the Conjugate Gradient method that seems well suited for massively parallel architectures. However now we have to implement this algorithm on a massively parallel machine with a few hundreds of processors. This work is underway. But, there are still some questions that remain to be solved. What is the best strategy between keeping entries by value or by position? What is the best way to solve the problem on the coarse mesh? Do approximate inverses perform better than the DPCG solver for M_{CC}? These questions are being studied and will be the subject of a forthcoming paper.

References

1. M. Dryja, W. Proskurowski and O. Widlund, *Numerical experiments and implementation of a domain decomposition method with cross points for the model problem*, Advances

in Computer Methods for Partial Differential Equations VI (R. Vichnevetsky and R.S. Stepleman, eds.), IMACS, 1987, pp. 23–27.
2. M. Haghoo and W. Proskurowski, *Parallel implementation of a domain decomposition method*, Technical Report, CRI 88–06 (1988).
3. W. Proskurowski and S. Sha, *Performance of the Neumann-Dirichlet preconditioner for substructures with intersecting interfaces*, Domain Decomposition Methods for Partial Differential Equations (T.F. Chan, R. Glowinski, J. Périaux and O. Widlund, eds.), Siam, 1990, pp. 322–337.
4. Barry F. Smith, *A domain decomposition algorithm for elliptic problems in three dimensions*, Numer. Math. **60** (1991), 219–234.

COMMISSARIAT À L'ENERGIE ATOMIQUE, CENTRE D'ETUDES DE LIMEIL–VALENTON, 94195 VILLENEUVE ST GEORGES, FRANCE

E-mail address: ciarlet@etca.fr

COMMISSARIAT À L'ENERGIE ATOMIQUE, CENTRE D'ETUDES DE LIMEIL–VALENTON, 94195 VILLENEUVE ST GEORGES, FRANCE

E-mail address: meurant@etca.fr